IN THE CLAIMS:

1. (currently amended) An object oriented computing system on a computer platform, comprising:

objects comprising software components which are dynamically loadable at runtime and which have ~~with~~ dynamically linkable named inputs and outputs stored on a memory of the computer system, said components also having internal tasks for queuing of data transferred into and out from the components via said inputs and outputs; and

an event communication framework providing automated, pattern-based, fully distributable events such that when a new dynamically loadable at runtime software component is loaded into said computer system also having dynamically linkable named inputs and outputs, the new software component inputs and outputs are all automatically linked to the inputs and outputs of the same name of said stored software components, so that the new and stored software components are combined ~~substantially~~ without changing any code within the software components and without writing any adapters.

2. (original) The object oriented computing system of claim 1, wherein the inputs and outputs of the objects are provided via CsaConnectable and CsaRemote objects, respectively.

3. (original) The object oriented computing system of claim 2, wherein each data structure associated with the inputs and outputs is described in a separate header file which can be used by every object to be linked.

4. (currently amended) The object oriented computing system of claim 2, wherein each object is a shared library which is dynamically ~~linkable~~ loadable at

runtime by an ASCII configuration ~~filing~~ file containing the names of the named inputs and outputs of the objects.

5. (currently amended) An object oriented computing system on a computing system, comprising:

a memory of the computing system storing objects;

said objects comprising software components which are dynamically loadable at runtime and having dynamically linkable named inputs and outputs and internal tasks for queuing of data transferred into and out from the objects via said inputs and outputs, respectively; and

an event communication framework providing automated, pattern-based, fully distributable events such that when a new software component is loaded into said computer system also having dynamically linkable named inputs and outputs, the new software component inputs and outputs are all automatically linked to the inputs and outputs of the same name of said stored software components so that the new and stored software components are combined ~~substantially~~ without changing any code and without writing any adapters.

6. (original) The object oriented computing system of claim 5, wherein the inputs and outputs of the objects are provided via CsaConnectable and CsaRemote objects, respectively.

7. (original) The object oriented computing system of claim 6, wherein each data structure associated with the inputs and outputs is described in a separate header file which can be used by every object to be linked.

8. (currently amended) The object oriented computing system of claim 6, wherein each object is a shared library which is dynamically ~~linkable~~ loadable at

runtime by an ASCII configuration file containing the names of the named inputs and outputs of the objects.

9. (currently amended) A method for designing software components in an object oriented computing system, comprising the steps of:

defining input and output events that are fully distributable;

configuring ~~dynamic linkable~~ software components which are dynamically loadable at runtime by dynamically linkable named input and output ~~connections~~ connection points and storing the components on a memory of the computer system, said components also having internal tasks for queuing of data transferred into and out from the components via said input and output connection points; and

providing autorouted pattern based fully distributable events based on an event communication framework such that when a new dynamically loadable at runtime software component is loaded into said computer system also having dynamically linkable named ~~inputs and outputs~~ input and output connection points, the new software component ~~inputs and outputs~~ input and output connection points are all automatically linked to the ~~inputs and outputs~~ input and output connection points of the same name of said stored software components, so that the software components are combined ~~substantially~~ without changing any code within the software components and without writing any adapters.

10. (currently amended) A storage medium including object oriented code having an object oriented computing system on a computer platform, comprising:

objects comprising software components which are dynamically loadable at runtime ~~with~~ and having dynamically linkable named inputs and outputs stored in memory of the computer system, said components also having internal tasks for

queuing of data transferred into and out from the components via said inputs and outputs; and

an event communication framework providing automated, pattern-based, fully distributable events such that when a new dynamically loadable at runtime software component is loaded into said computer system also having dynamically linkable named inputs and outputs, the new software component inputs and outputs are all automatically linked to the inputs and outputs of the same name of said stored software components, so that the software components are combined ~~substantially~~ without changing any code within the software components and without writing any adapters.

11. (original) The storage medium of claim 10, wherein the inputs and outputs of the objects are provided via CsaConnectable and CsaRemote objects, respectively.

12. (original) The storage medium of claim 11, wherein each data structure associated with the inputs and outputs is described in a separate header file which can be used by every object to be linked.

13. (currently amended) The storage medium of claim 12, wherein each object is a shared library which is dynamically ~~linkable~~ loadable at runtime by an ASCII configuration ~~filing~~ file containing the names of the named inputs and outputs of the objects.

14. (currently amended) A storage medium, comprising:

object oriented code for an object oriented computing system on a computing system;

objects comprising software components which are dynamically loadable at runtime stored on a memory of the computer system and having dynamically linkable

named inputs and outputs and internal tasks for queuing of data transferred into and out from the objects via said inputs and outputs, respectively; and

an event communication framework providing automated, pattern-based, fully distributable events such that when a new software component is loaded into said computer system also having dynamically linkable named inputs and outputs, the new software component inputs and outputs are all automatically linked to the inputs and outputs of the same name of said stored software components, so that the software components are combined ~~substantially~~ without changing any code of the software components and without writing any adapters.

15. (original) The storage medium of claim 14, wherein the inputs and outputs of the objects are provided via CasConnectable and CsaRemote objects, respectively.

16. (original) The storage medium of claim 15, wherein each data structure associated with the inputs and outputs is described in a separate header file which can be used by every object to be linked.

17. (currently amended) The storage medium of claim 15, wherein each object is a shared library which is dynamically ~~linkable~~ loadable at runtime by an ASCII configuration file containing the names of the named inputs and outputs of the objects.

18. (currently amended) A method for designing software components in an object oriented computing system having a storage medium including object oriented code, comprising the steps of:

defining input and output events that are fully distributable;

configuring ~~dynamic linkable~~, software components which are dynamically loadable at runtime by dynamically linkable named input and output ~~connections~~

connection points and stored on a memory of the computer system, said components also having internal tasks for queuing of data transferred into and out from the components via said input and output connection points; and

providing autorouted pattern based fully distributable events based on an event communication framework such that when a new software component is loaded into said computer system also having dynamically linkable named ~~inputs and outputs~~ input and output connection points, the new software component ~~inputs and outputs~~ input and output connection points are all automatically linked to the ~~inputs and outputs~~ input and output connection points of the same name of said stored software components, so that the software components are combined ~~substantially~~ without changing any code within the software components and without writing any adapters.